Modern Computational Accelerator Physics

James Amundson Alexandru Macridin *Panagiotis Spentzouris*

Fermilab

USPAS January 2015

2D Rectangular space charge solver

2D Solvers

- 2D solvers are fast (compared with the more realistic 3D solvers).
- The beam density is assumed to be longitudinally uniform or to be weakly longitudinal dependent.
- The Poisson equation reduces to

$$\frac{\partial^2 \Psi}{\partial x^2} + \frac{\partial^2 \Psi}{\partial y^2} = -\frac{\rho(x, y)}{\epsilon_0}$$

- Different solvers calculate the potential employing different approximations.
- It produces only transverse kicks.

Particle In Cell (PIC) Solvers

- Grid based techniques
- The PIC solvers follow the following steps:
  - The charge is deposited on a numerical grid.
  - The electric field is calculated on the grid.
  - The electric field is interpolated from the grid at the particle position.

Conducting rectangular boundary

- The beam pipe is rectangular and made from a perfect conducting material.
- These assumptions imply

$$\Psi(x=0,y)=\Psi(x=L_x,y)=0$$
$$\Psi(x,y=0)=\Psi(x,y=L_y)=0$$

  where the pipe horizontal and vertical dimensions are L_x and respectively L_y.

- The numerical grid in our solver covers the transversal cross-section of the pipe.

Charge deposition

- Define the grid.
- Deposit the particle on the four nearest grid points (*Cloud in Cell method*).
  - If the grid point is at distance (offset_x, offset_y) from the particle, deposit the weight (1-offset_x)*(1-offset_y).
    - offset_x and offset_y are scaled by the grid cell size.
  - This is not the only possible way to deposit charge on a grid.

Assignment 1

- Run the script bunch.py and try to understand it.
  - The script bunch.py creates a gaussian bunch with a given covariance.
  - The bunch object is imported from synergia.
  - Synergia functions used are:
    - Reference_particle(proton_charge, mass, total_energy)
    - Bunch(reference_particle, numbers_of_macroparticles, real_number_of_protons, comm)
    - populate_6d(dist, bunch, means, covariance_matrix)
    - print_matched_parameters(Cmat,beta, bunch_number)

- The script charge_deposit.py deposits the beam charge on a rectangular grid.

Run the script and try to understand it. Increase the grid number of points. Notice that for a very fine grid a charge deposition which goes beyond the four nearest grid points would produce a smother distribution.

Poisson equation

- The Poisson equation can be solved in the Fourier space.
- The potential (or any function which vanish on a rectangular boundary) can be written as

$$\Psi(x,y) = \sum_{m,n>0}^{\infty} \Psi_{mn} sin\frac{\pi mx}{\text{\L}_x} sin\frac{\pi ny}{\text{\L}_y}$$

where

$$\Psi_{mn} = \frac{4}{L_x L_y} \int_0^{L_x} dx \int_0^{L_y} dy \Psi(x,y) sin\frac{\pi mx}{\text{\L}_x} sin\frac{\pi ny}{\text{\L}_y}$$

In the Fourier space the Poisson equation is

$$\left(\frac{\pi^2 m^2}{{L_x}^2} + \frac{\pi^2 n^2}{{L_y}^2}\right)\Psi_{mn} = \frac{\rho_{mn}}{\epsilon_0}$$

Electric field calculation

- The electric field E_x on the grid is given by

$$E_x(x,y) = -\frac{\Psi(x+h_x,y)-\Psi(x-h_x,y)}{2h_x}$$

where h_x is the grid cell size.

- Analogous expression for E_y.

Assignment 2

- Write a python script which calculates the electric potential on a grid with zero rectangular boundary conditions. For the charge deposition use charge_deposit.py.
  - Use *synergia.foundation.pconstants.epsilon0* for ϵ_0
- Make a 3D plot of the potential.

Assignment 3

- Calculate the electric fields E_x and E_y on the grid.
- Make 3D plots of E_x and E_y.